ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 12, December 2021

True Random Number Generation Using Programmable Delays

[1],*Manish and [1]Dr Manisha Bharti

[1]Electronics & Communication and specilization in VLSINatinal Institute of Technology, Delhi, India
*corresponding author: Email: 2221007@nitdelhi.ac.in

Abstract*— To meet the demand of better cryptographic systems a TRNG (True random number generator) play a vital role. This paper shows an efficient method on FPGA that utilizes free running oscillators in which random jitter is used as a source of randomness for the generation of true random numbers. Programmable delay lines used to generate variations and to add jitter in the free running oscillator rings. This true number generator has advantage over previous designs as it reduces the correlation for the equal length ring oscillators which leads to multiple zeros in the random numbers using the programmable delay lines. The paper is implemented on Xilinx Artix-7 FPGA.

***Index Terms*— TRNG, jitter, PDL, ring oscillator**

I. INTRODUCTION

As we know random number have wide range of applications in real world like cryptographic system, key generation, random bit padding, lottery drawing, gambling, probabilistic algorithms and computers games, A true random number generator can be used to accomplish them. A true number should be non-deterministic and unpredictable because poor random sequence is prone to threats like PRNG in which secret key for the smart cards can be retrieved [1], which can be tackled with the help of TRNGs. Diehard [2] and NIST [3] statistical tests are used to check the randomness of sequence while the stochastic model [4] is used entropy per-bit for the randomness. Entropy and postprocessing operation of the TRNGs are generally from a single source like metastability [6], [7], thermal noise [5], clock jitter [8], [9] and chaos [10]. First the physical noise is used as the source of randomness to generate the raw random bit-stream using a sampler, then to improve the entropy (randomness) the postprocessing operations like Neumann corrector [11] or hash function [5] are used in TRNG. There are various FPGA prototypes of TRNG with post processing designs have proposed in which entropy is drives from jitter of Ring Oscillator (RO) [12],[13] or setup or hold time violations of flip-flops (FFs) that causes the metastability of flip-flops [6], [7]. While implementing the oscillator ring base TRNG there are different challenges arises like the lesser entropy in random sequence because of high correlation of equal length ring oscillators as they have identical delays. This problem can be solved with the help of PDLs (programmable delay lines) in the oscillator ring as they cause jitter in the RO clocks and creates higher variations in the RO ring. The PDLs are incorporated in each sampling clock of ring oscillator which increases the randomness as the output of ring oscillators are not correlated with each other. After getting the random sequence from PDL based ROs the post processing block that is the Von Neumann corrector is used which removes the bit bias from the random sequence and produces the higher random sequence. The work is implemented on the Xilinx Artix-7 FPGAs (xc7a35tcpg236-1).

II. COMPONENTS OF TRNG

A. Programmable Delay Lines (PDLs)

The modifications in the LUT's propagation delays in different inputs can be used to generate the internal variation of FPGA Look-Up Tables (LUT's) [6].

For example, the figure 1 shows a 4 PDL input LUT which is programmed to implement an inverter which inverters the input A_1, and other inputs A_2, A_3 and A_4 will act as "don't-care" bits but they affect the signal propagation path from A1 to the output. As shown in figure. 1, the signal propagation path (A1 to the output) is shortest for $A_2\ A_3\ A_4 =$ 000 (indicated with solid red line) and longest for $A_2\ A_3\ A_4 =$ 111 (indicated with dashed blue lines) for 4-input LUT's. In this way using a single LUT a programmable delay inverter with three control inputs can be implemented in which A_1 is the inverter input and other 3 inputs ($A_2\ A_3\ A_4$) are used to control 2^3=8 discrete levels.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 12, December 2021

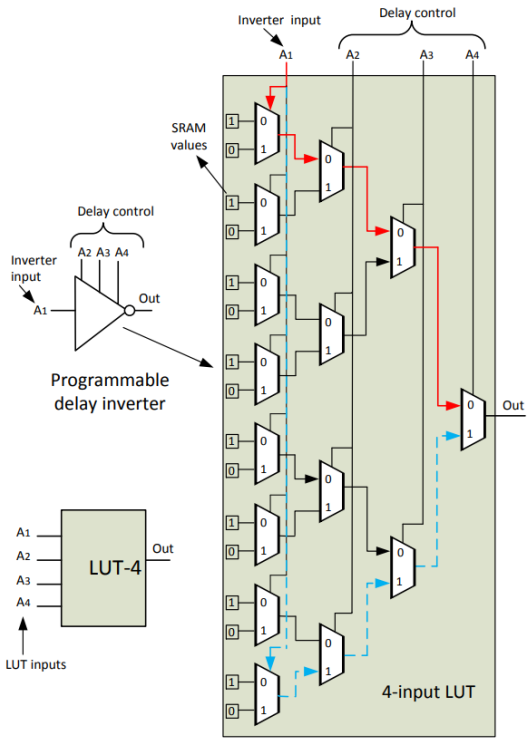

Fig. 1. using a 4-input LUT

B. Ring Oscillator Based TRNG

As discussed earlier that there are number of RO based TRNG designs [12]-[18] in which odd number of delay elements (inverters) are used in ring configuration and jitter is added in the free running oscillator rings. This produces digital value of the oscillator output in which the period is approximately *2TN* where T represents the delay of one inverter and N represents the number of inverters in ring. From cycle-to-cycle variation in the period causes the jitter at the rising and falling edges of the generated ring oscillator clocks as represented in figure 2.

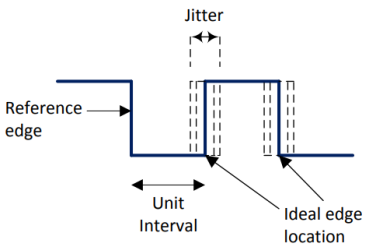

Fig. 2: Jitter in clock

A jitter can be generated with these oscillations, also in digital circuits it can be occurred due to various sources like power supply variations, cross talks, temperature variations, semiconductor noise, and propagation delays. Truly random bits can be generated using such jitters and D flip flop (DFF) based sampler by sampling high frequency RO output as shown in figure 3.

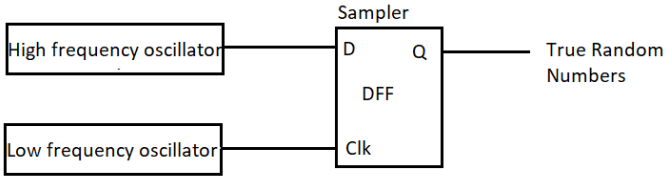

Fig. 3. Basic oscillator-based

Utilising multiple free running ROs can be used to improve the randomness of random bits generation [12] which can be achieved by XORing the outputs of ROs and then sample them with a fixed frequency reference clock using a D flip flop for the generation of a true random bit as illustrated in figure 4.

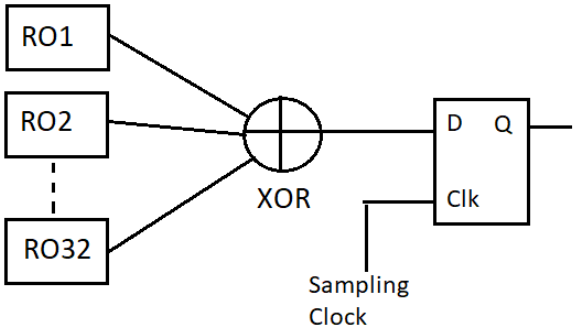

Fig. 4 Block diagram of the Sunar type TRNG

But it is a very difficult to handle high number of switching activity XORing using XOR block and sampling DFF from the free-running ring oscillators [13] because of the high transitions during a sampling because of the parallel ring oscillators place severe setup and hold-time requirements. This problem can be tackled to some point by using Wold type [15] setup in which an extra DFF is incorporated with each free-running ring oscillator's output as in figure 5.

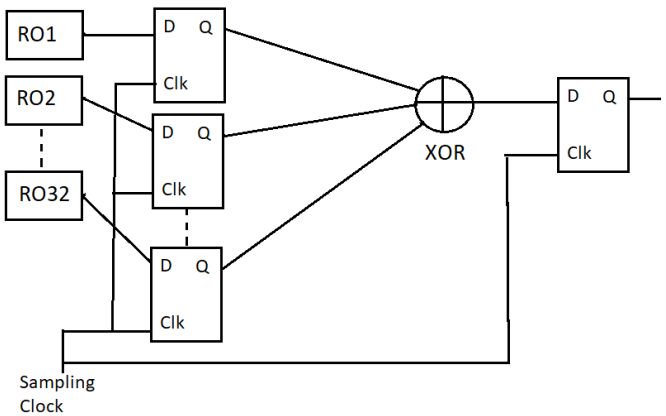

Fig. 5 Block diagram of the modified TRNG (Wold type)

Random sequence produced by this design can pass the NIST statistical tests even without post processor and uses lessened number of ROs. But, this design having the mathematical

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 12, December 2021

complexity like in the original design [12] like mutual dependence of rings [17] which results in lack of randomness in the output sequence. The entropy can be enhanced at the cost of lower throughput or high hardware resources [18].

III. TRNG ARCHITECTURE

Before that it is important that the ring oscillator based TRNGs having very low entropy when identical ring oscillator are in use [14]. The reason is the mostly zeros in the output of rings as the equal length rings having very high correlation with each other and this same delays in the XOR results in zeros at the output which results in poor random sequence. To overcome this problem a PDLs are incorporated in the TRNG in which the random jitter of free running oscillators is used for the generation of true random numbers. Large variation in oscillators rings and to add jitter in free running RO clocks PDL used. The main benefit of this design is reduction in the correlated equal length RO. For example, incorporating PDL for the variable ring oscillator outputs for sampling clock as shown in figure 6.

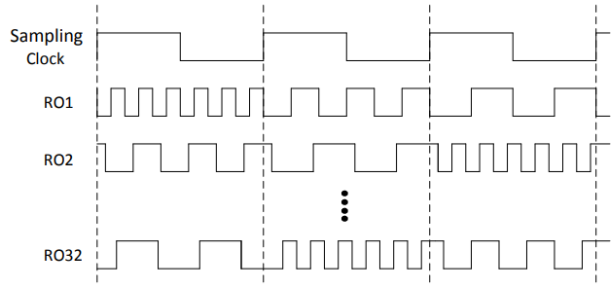

Fig. 6. RO outputs for each sampling clock by using PDL

Furthermore, cycle to cycle variation in RO oscillations are because of the inverter delay. The randomness improves when it XOR. The implementation of TRNG is done on Xilinx Artix-7 FPGAs using Xilinx Vivado, suite 14.1 software.

A. Design Overview

The architecture of true random number genrator is shown in the figure 7. Here, 3 inverters and 1 AND gate is used to realize a RO that is marked by black boxes. Each AND gate only use to enable the respective RO. To design a RO four LUTs are required 3 for inverter and 1 for the AND gate on FPGA. So as to generate PDL inside the 4-input LUT, one input is use as the inverter input in ring oscillator while the other 3 inputs are used to configure 2^3=8 discrete levels.

The TRNG architecture have 32 RO's, XOR block, D flip flops. The 'enable' input of control circuitry starts the all 32 ROs simultaneously. Then the outputs of each RO are sampled at 24 MHz sampling frequency using a DFF. Then these outputs are merged by a XOR tree and again at the frequency of 24 MHz it sampled using a DFF. Then for incorporating PDLs in each clock cycle 2^3 discrete levels are randomly used for the delay control inputs. Then the raw random bitstream directly sent to 8-bit shift register using a multiplexer and a control signal. Finally, byte stored in FIFO and taken through a USB interface.

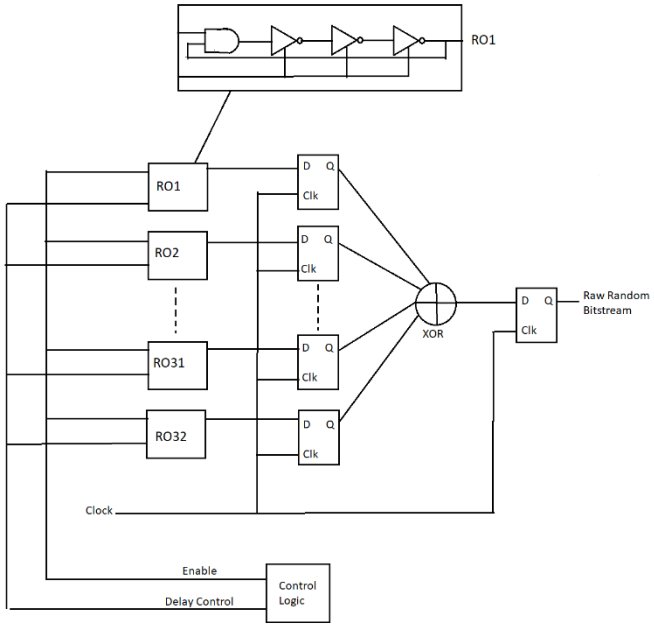

Fig. 7. TRNG architecture

B. Post-Processing

The post processing block is established using the Von Neumann corrector [11] that is used to improve the randomness by bit bias removal in the generated random bit output sequence. The basic operation of Von Neumann corrector is illustrated in figure 8. It reads two bits of TRNG random sequence at same time and discards the same bits (i.e., 11 and 00) and if they are not same (i.e., 10 and 01) it takes the first bit and discards the second bit. For 512 bits of raw input on an average post-processor generates 128 bits of output.

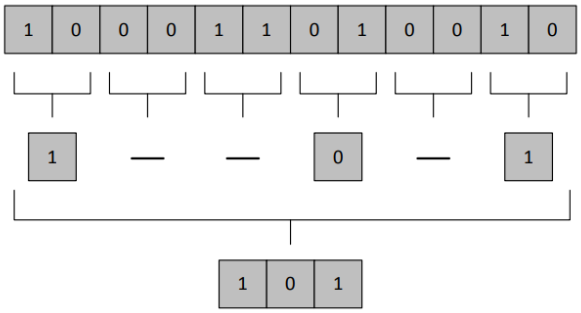

Fig. 8. Principle of operation of the Von Neumann

IV. DESIGEND TRNG ARCHITECTURE

A. Ring oscillator design and results

A basic Ring oscillator design is shown in figure 9. RO's in the implemented TRNG having one AND gate and three inverters that uses PDLs to introduce jitter in the RO's clock.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 12, December 2021

Here, 'En' is input control signal to enable the ring oscillator and 'a' is the 3-bit control signal that uses to provide 2^3 discrete levels to the inverter.

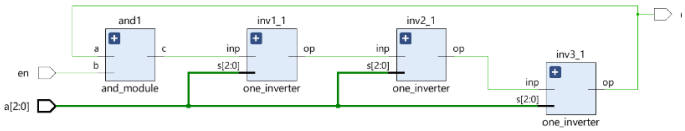

Fig. 9: Ring oscillator

Here each inverter has 4 inputs in which 3 inputs are used for incorporating PDL and one inverting input. The internal view of an inverter is shown in figure 10. Here each inverter is designed using 8 inverters with different propagation delays and a 8:1 multiplexer whose select line are 3-bit the control signal 'a'. The values assigned to 'a' are arbitrary so it produces true random sequence at the final output.

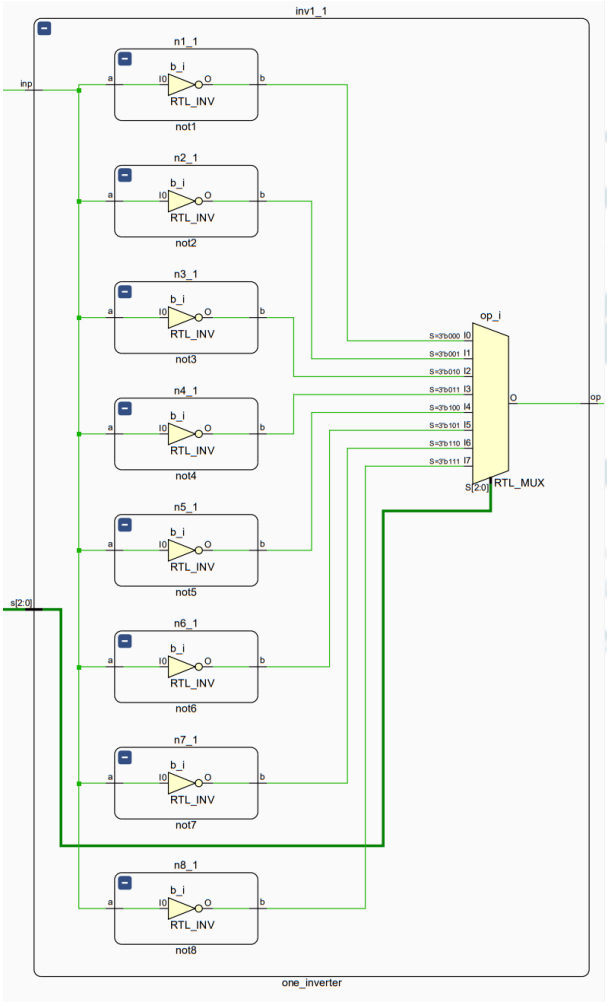

Fig. 10: inverter using a PDL

To design the inverter of different delays the inbuilt logic gate is used as their propagation delay can be synthesis using the EDA tools. So, the output of a ring oscillator after incorporating PDLs is shown in figure 11.

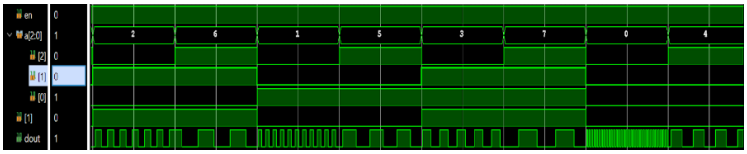

Fig. 11: Output of a ring Oscillator

Here, we can clearly see that for the different values of control signal 'a' the RO clock frequency changes like for 'a'=000 it provides shortest path that means it provides higher clock frequency and for the case 'a'=111 it provides the longest path that means the lower clock frequency.

After implementing the RO's the next step is to implement 32 RO's in which for each ring the delays provide by PDL using select line 'a' should be different i.e., for case a=000 may provide shortest path in first ring but for the second ring same value of 'a' may provide the longest path and for the third ring it may provide some intermediate path. For that there are 40320 cases for the 32 RO's. For handling this problem, the random 32 cases out of 40320 cases are taken. So, after implementing that the RO clock of different RO's for each cycle is shown in figure 12 for 10 RO's.

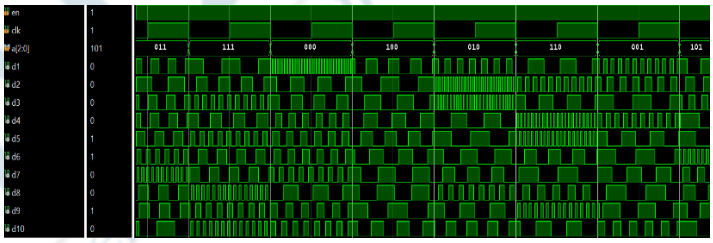

Fig. 12: ring oscillator outputs for each sampling clock by using PDL

Here, it can clearly notice that for the same control signal 'a' each RO's provides the different clock frequency which reduces the problem of basic TRNG [12] in equal length RO's having the lower entropy issue. After the RO's the output of each RO's is sampled using a DFF as in WOLD-type design [15]. Then it will be combined using a XOR tree and then again sampled through a DFF to provide the final raw true random bit stream as output. The final design of the TRNG is shown in figure 14.

B. TRNG design and results

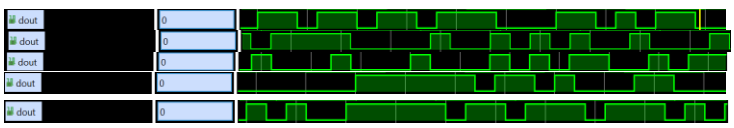

Fig. 13: Random Sequence at different instants

Figure 13 shows the true random sequence bit stream generated using the implemented TRNG. The figure shows the random sequence at the different instant of times.

ISSN (Online) 2394-6849

International Journal of Engineering Research in Electronics and Communication Engineering (IJERECE)

Vol 8, Issue 12, December 2021

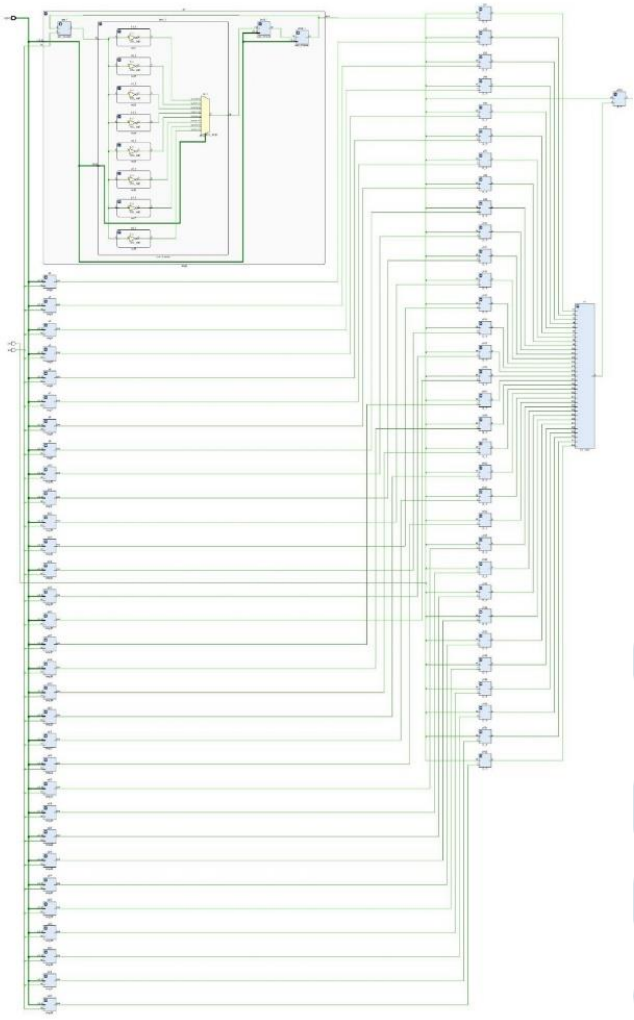

Fig. 14: TRNG Design

C. Post processing block results

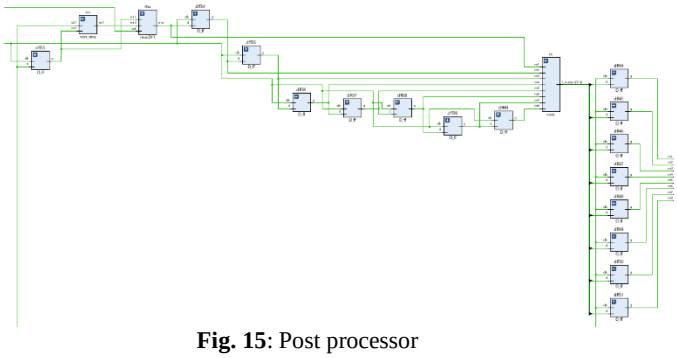

Fig. 15: Post processor

Here the first block is of post processing block Von Neumann corrector which reduces the bit bias in the output bit stream, then there is a block of 2x1 multiplexer in which one input is raw bit stream and the second one is post processed bit stream and based on select line the bit stream can be selected and gives the 8-bit random numbers.

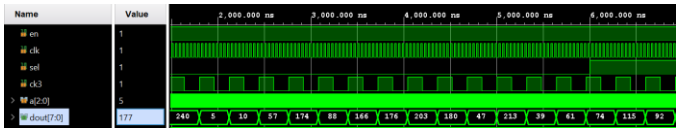

Fig. 16: TRNG with raw bit stream

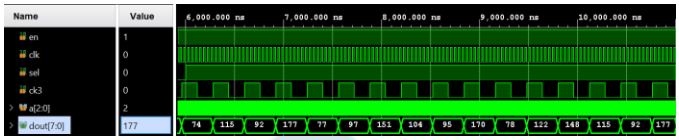

Fig. 17: TRNG with post processor

The simulation results show the random output on the basis of select line sel, when it is at low level it gives the random number using the raw bit stream and when it is at high level it gives post processed random numbers.

D. Synthesis results

Table 1. Synthesis results

Resource	Used	Available	Utilization
LUT	41	20800	0.20
FF	48	41600	0.12
IO	12	106	11.32
BUFG	2	32	6.25

Table 1 shows the synthesis results of the implemented TRNG design, for the TRNG design its utilization is very less as it only uses 41 LUTs, 48 Flip flop & latches, 12 IOs and 2 BUFG.

V. Conclusion

As we know random number have wide range of applications in real world like cryptographic system, key generation, random bit padding, lottery drawing, gambling, probabilistic and computers games, A true random number generator can be used to accomplish them. The ring oscillator based TRNG have some issues when the equal length rings used which having the identical delays, to solve this issue incorporated the PDLs to introduce in the clocks of the free running RO's. This will enhance the quality of the true random bit stream. And adding the post processing block removes the bit biasness and enhance the entropy of random numbers.

REFERENCES

[1] N. Nalla Anandakumar; Somitra Kumar Sanadhya; and Mohammad S. Hashmi. "FPGA-Based True Random Number Generation Using Programmable Delays in Oscillator-Rings" in IEEE Transactions on Circuits and Systems II: Express Briefs (Volume: 67, Issue: 3, March 2020), pp 570-574.

[2] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control," in Cryptographic Hardware and Embedded Systems – CHES 2011. Springer Berlin Heidelberg, 2011, pp. 17–32.

[3] K. Wold and C. H. Tan, "Analysis and Enhancement of Random Number Generator in FPGA Based on Oscillator Rings," in Int. Conf. on Reconfigurable Computing and FPGAs, Dec 2008, pp. 385–390.

[4] J. von Neumann, "Various techniques used in connection with random digits," in Monte Carlo Method. National Bureau of Standards Applied Mathematics Series, 12, 1951, pp. 36–38.

[5] K. Wold and S. Petrovi, "Security properties of oscillator rings in true random number generators," in 2012 IEEE 15th International Symposium on Design and Diagnostics of Electronic Circuits Systems (DDECS), April 2012, pp. 145–150.

[6] B. Sunar, W. J. Martin, and D. R. Stinson, "A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks," IEEE Transactions on Computers, vol. 56, no. 1, pp. 109–119, Jan 2007.

[7] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control," in Cryptographic Hardware and Embedded Systems – CHES 2011. Springer Berlin Heidelberg, 2011, pp. 17–32.

[8] A. Maiti, R. Nagesh, A. Reddy, and P. Schaumont, "Physical Unclonable Function and True Random Number Generator: A Compact and Scalable Implementation," in Proceedings of the 19th ACM Great Lakes Symposium on VLSI, ser. GLSVLSI. ACM, 2009, pp. 425–428.

[9] W. Schindler and W. Killmann, "A proposal for: Functionality classes for random number generators," 2011.

[10] M. Majzoobi, F. Koushanfar, and S. Devadas, "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control," in Cryptographic Hardware and Embedded Systems – CHES 2011. Springer Berlin Heidelberg, 2011, pp. 17–32.